Report on the DNA/Biomolecular Computing Workshop*

Arthur L. Delcher
Department of Computer Science

Lee Hood
Department of Molecular Biotechnology
University of Washington

Richard M. Karp
Department of Computer Science and Engineering
University of Washington

June 6-7, 1996

Abstract

This report is a summary of the DNA/Biomolecular Computing Workshop held on June 6-7, 1996 in Arlington, VA. The purpose of the workshop was to bring together researchers from both computer science and biotechnology to assess the current state of biomolecular computing, and to identify and explore the most promising approaches and most critical problems of this technology. Presentations by both computer scientists and biologists described the current state and future directions of biomolecular computing research and of competing computing technologies. The consensus of the participants was that although there is no clear road map toward achieving an efficient biomolecular computing device, there are many research directions that should be pursued that could lead to such a device.

Executive Summary

The current explosion of interest in molecular computing was sparked by Adleman's 1994 *Science* paper in which he showed how to use DNA to encode and solve a "toy" 7-city travelling salesperson problem. Although 7 cities is such a small problem that it can be solved at first glance by anyone, the method used could in principle be extended to much larger instances of the problem. This problem is particularly interesting since it is a member of an important class of problems known as *NP-complete* for which there are no known efficient algorithms on

---

* This report was written by the participants of the workshop and produced by S. Rao Kosaraju. The workshop was supported by the National Scienece Foundation through Grant No. CCR-9642739 to the Johns Hopkins University. However, any opinions, findings conclusions, and recommendations expressed herein are those of the authors and do not necessarily reflect the views of NSF.

conventional computers. The reasons for optimism about this approach were the minute amounts of DNA and energy used and the large number of molecules that could be operated on concurrently. But questions remained about the scalability of the approach and the cost of materials for it.

Since then, much work has been done, both in assessing the theoretical computational issues of molecular computing as well as studying and improving the practical aspects of the biochemical systems. Molecular computing has been shown to be universal, meaning that it is theoretically capable of performing any computation that a conventional computer can. Several combinations of computational primitives, supported by a variety of biochemical reactions, have been proposed, and some have been tested in the laboratory.

The DNA/Biomolecular Computing Workshop was held on June 6-7, 1996 in Arlington, VA. It was designed to bring together researchers from both computer science and biotechnology to assess the current state of biomolecular computing, and to identify and explore the most promising approaches and most critical problems of this technology. The format of the workshop consisted of informal presentations alternating with lively open discussions. On the last afternoon of the workshop, separate working groups of computer scientists and of biologists met and prepared lists of research areas to be pursued. In the final session these lists were presented and discussed.

The consensus view was that it is unlikely that a general-purpose biomolecular computer capable of outperforming a conventional electronic computer will be produced in the near future. This field is in its infancy, however, and more basic research needs to be done to assess its capabilities. There are many possibilities for molecular computing architectures that should be explored. Many of these, in addition to being computationally interesting, will almost certainly generate useful technologies for non-computational applications. Specific research areas that should be pursued include better characterization of the chemical reactions involved as computational primitives, improved or alternative techniques for performing input/output, and new computational models based on molecular components other than DNA in solution.

1 Introduction

A series of discoveries over the past fifty years have illuminated the extraordinary capabilities of living cells to store and process information. We have learned that genes encoded digitally as nucleotide sequences serve as a kind of instruction manual for the chemical processes within the cell and constitute the hereditary information that is passed from parents to their offspring. Information storage and processing within the cell is more efficient by many orders of magnitude than electronic digital computation, with respect to both information density and energy consumption.

The field of recombinant DNA technology has developed, based on procedures for synthesizing, cutting, splicing, copying, replicating and reading DNA molecules, and for separating and classifying them according to their size or content. These processes are fairly slow but highly parallel, operating on as many as 10^{16} molecules at a time. We have the ability to

create designer genomes by splicing together selected fragments from different organisms, and cloning them by exploiting the self-replicating ability of bacterial cells. Recombinant DNA technology has also led to the creation of DNA hybridization arrays that can be used to analyze genomes and detect mutations associated with disease.

Recombinant DNA technology is an example in which the information processing capabilities of biological molecules are exploited for technological purposes. Another example is combinatorial chemistry, in which proteins with a specific activity, such as binding to a particular antigen, are synthesized by an iterative process which operates on a large pool of candidate proteins simultaneously *in vitro*, alternately selecting those proteins that are fittest with respect to the desired activity and then mutating them randomly to obtain a new pool of candidate proteins.

In the examples of recombinant DNA technology and combinatorial chemistry, processes inherent to living cells are used to analyze or modify biological molecules, and to select those with desirable properties. The field of biomolecular computing goes further by using these processes to perform information processing that has no intrinsic connection with the processing of biological molecules - in other words, biological molecules are used as a medium for general-purpose digital computation.

Biomolecular computing leapt into prominence in late 1994 through the work of Len Adleman, who performed a laboratory experiment in which a collection of DNA molecules was constructed representing the possible solutions to a toy combinatorial problem, and recombinant DNA techniques were used to sift through these molecules to select the correct solution. Subsequent work has introduced many refinements, but has for the most part stuck to Adleman's original generate-and-test approach, in which a large collection of candidate solutions is generated, and tests are then performed in order to discard the incorrect solutions, until only the correct ones remain.

The original enthusiasm, based on the extraordinary energy efficiency and compactness of information storage afforded by the DNA medium, together with the extraordinary degree of parallelism of recombinant DNA procedures, is tempered by a number of sobering concerns. Among these are the following:

- It seems infeasible to operate on more than about 10^{16} DNA molecules at once. Thus the generate-and-test approach is limited to problems with at most 10^{16} candidate solutions. Typical problems of this size are easily dealt with by conventional methods of computation, leaving little advantage for DNA computing.

- DNA processing is slow, so that long sequences of steps must be avoided.

- DNA processing is error-prone. Error-correction techniques are available, but they multiply the numbers of steps required and the numbers of molecules processed in each step.

- In a large-scale DNA computation the cost of materials such as oligos, ligases, polymerases,

restriction enzymes and hybridization enzymes may be prohibitive.

Because of these considerations DNA computation as we currently understand it may have a very limited range of application. Certainly no "killer application" has been identified yet. Nevertheless, biomolecular computing is still in its infancy. Our understanding of it, currently limited to a particular paradigm of DNA computing, will surely broaden as we gain a better understanding of information storage and processing in living cells, and the challenge of achieving cost-effective biomolecular computing will surely serve as a forcing function for advances in both biotechnology and computer science. Despite the short duration of our workshop, a number of promising, albeit speculative, directions for investigation were identified. We conclude the Introduction by mentioning some of them.

Information Processing Mechanisms in the Cell There is a great deal of science to be done in elucidating the mechanisms by which living cells store and process information and developing new biochemical tools and techniques based on these mechanisms. These new mechanisms will inevitably suggest new modes of biomolecular computing. Examples of such possibilities include:

- ✓ using artificial analogs of DNA and protein as computing media;
- ✓ using DNA to form self-assembling two- and three-dimensional structures analogous to cellular automata;
- ✓ exploiting the information inherent in the secondary and tertiary structure of DNA and proteins;
- ✓ using living cells as components in computing systems.

We should investigate **Novel Computer Architectures**, such as analog biomolecular computers and hybrid computers containing general-purpose electronic processors interacting with special-purpose biomolecular processors, as well as **Special Applications** such as sequence assembly or drug design, in which the information to be operated upon is inherently chemical or biochemical, so that costly input/output conversions between electronic and biochemical media can be dispensed with.

Departures from the Generate-and-Test Paradigm Combinatorial problems can be attacked by parallel local search algorithms, which maintain a large but not exhaustive pool of potential solutions, and gradually improve the quality of the pool by mechanisms of selection, crossover and replication analogous to those used in combinatorial drug design. DNA implementations of such procedures would not require the generation of all possible solutions, and could therefore be applied to larger problem instances.

Spin-offs The development of DNA computing will require the refinement and careful characterization of basic procedures for separating, ligating, cutting and amplifying DNA. The improved implementation and understanding of these procedures should find important applications in the biotech industry.

2 Participants

The workshop was held June 6-7, 1996, at the National Science Foundation in Arlington, VA. Invited participants were:

Lee Hood, Department of Molecular Biotechnology, University of Washington, Co-chair

Richard M. Karp, Department of Computer Science and Engineering, University of Washington, Co-chair

Leonard Adleman, Department of Computer Science, University of Southern California (teleconference)

Tilak Agerwala, Director, Parallel Architecture and Systems Design, IBM, Poughkeepsie, NY

Gary Benson, Department of Biomathematical Sciences, Mt. Sinai School of Medicine

Nickolas Chelyapov, Laboratory for Molecular Science, University of Southern California

Anne Condon, Department of Computer Science, University of Wisconsin, Madison

Arthur L. Delcher, Department of Computer Science, Loyola College in Maryland

James C. Ellenbogen, Nanosystems Group, Mitre Corporation

David Gifford, Laboratory for Computer Science, Massachusetts Institute of Technology

Susan Hardin, Department of Biology, University of Houston

S. Rao Kosaraju, Department of Computer Science, Johns Hopkins University

Robert Lipshutz, Affymetrix

Tom Knight, Artificial Intelligence Laboratory, Massachusetts Institute of Technology

John Reif, Department of Computer Science, Duke University

Lloyd Smith, Department of Chemistry, University of Wisconsin, Madison

Richard Superfine, Department of Physics, University of North Carolina
Granger Sutton, Institute for Genomics Research

Richard Watson, Advanced Information Technology Computation Organization, Lawrence Livermore National Laboratory

Erik Winfree, Computation and Neural Systems Program, California Institute of Technology

3 History and Current State of Biomolecular Computing

After opening remarks, the workshop began with a presentation via teleconference by Leonard Adleman, University of Southern California, on the history and current state of research in the area of DNA and biomolecular computing.

The historical origins of molecular computing can be found in the work of researchers from three academic arenas. Logicians Gödel, Church, Kleene and Turing showed that universal computation could be accomplished using only memory and simple calculation steps. Biologists Watson and Crick discovered how genetic information is encoded digitally in DNA and how enzymes could manipulate this information. Finally physicist Richard Feynman argued that there were no inherent physical laws to prevent building extremely small computers.

The current explosion of interest in molecular computing was sparked by Adleman's 1994 *Science* paper in which he showed how to use DNA to encode and solve a "toy" 7-city travelling salesperson problem. Although 7 cities is such a small problem that it can be solved at first glance by anyone, the method used could in principle be extended to much larger instances of the problem.

The travelling salesperson problem consists of a collection of cities with a designated set of one-way connections from one city to another. The goal is to determine whether there exists a sequence of connections that visits every city without going to the same city twice. What makes this problem particularly interesting is that it is a member of an important class of problems known as *NP-complete* for which there are no known efficient algorithms on conventional computers.

Adleman solved the problem by designing a separate strand of DNA for each possible connection from one city to another. Moreover, the strands were created so that they would bind together exactly when the destination of one connection matched the source of another. The actual DNA was then fabricated and mixed together so that all possible bindings would occur. From the result, molecules that represented solutions to the problem could be isolated by extracting those molecules that had the correct length and which contained the code segments for all the cities.

There were several reasons to be optimistic about the potential of DNA computing based on the characteristics of Adleman's experiment. The volume of DNA used was very small, only 100 microliters. The speed of computation, specifically the rate at which molecules combined, was very fast, approximately 10^{14} operations/second (a pentium chip performs about 10^{8} operations/second while a parallel supercomputer may perform as many as 10^{12} operations/second). The energy used for the computation was extremely small, only 2×10^{19} operations/joule (within a factor of 17 of the theoretical optimum at room temperature). Finally

the density of memory storage was very large, about 1 bit/cubic nanometer (about $l0^{12}$ times denser than conventional videotape).

On the other hand, there were some clear difficulties to the approach. As implemented, Adleman's approach would require oceans of DNA to scale up to large enough problems to be of interest. The error rates in the molecular processes in the computation were high, especially in comparison to those of conventional computers. The cost of the materials in the experiment was high (some of the enzymes cost 10^5 as much as gold). Finally, there was no clear computational problem that people wanted to solve for which the DNA approach appeared superior to conventional computers. In other words, there was no "killer application".

In the ensuing year further work has been done both in assessing the theoretical computational issues of molecular computing as well as studying and improving the practical aspects of the biomolecular systems themselves. In the theoretical domain it has been shown that molecular computing is universal, *i.e.*, it is capable of performing any computation that any other kind of computer can (but, of course, not necessarily as efficiently). In the biology domain, work has progressed in characterizing and improving the biochemical operations that can be performed, and in designing new architectures for biomolecular computers including different subsets of allowed operations and different methods of holding the DNA being used (attaching it to glass for example, rather than keeping it in solution). Work also has progressed in designing DNA sequences with desirable properties and in controlling the error rates inherent in the biochemical reactions.

One promising alternative molecular computing architecture is the *stickers* mode, currently being explored by Adleman. In this model, long memory strands of DNA are used together with shorter strands--stickers--which can anneal (attach) to a unique site on a long strand. An attached sticker represents a 1-bit, while the absence of a sticker represents a 0-bit. A test tube contains a collection of identical DNA memory strands, but with different stickers annealed.

There are three basic operations supported by the stickers model. The simplest is a merge operation in which the contents of two tubes are combined. The second is a separation operation based on the presence or absence of a sticker at a particular location. Two tubes are produced--one containing all complexes with the particular sticker attached, the other containing the remaining complexes. The third operation is a sticker-attach operation that anneals a specific sticker to all memory strands in a test tube.

The stickers model can be used to design a system to break Data Encryption Standard (DES) encryption. Specifically, given a message and its encrypted version, the system will compute which 56-bit key was used for the encryption. The proposed system would use a rack of test tubes containing a total of 1.4 grams of DNA, manipulated by 32 robots under microprocessor control. It would require separation error rates of less than 1 bad molecule for every 10,000 good ones, and would use no expensive enzymes. If stickers operations could be performed at the rate of 1 per hour, then DES could be broken in 7 months; at the rate of 1 operation per second, DES could be broken in 1.5 hours.

There are several reasons to be optimistic about DNA computing in the future. It realizes massive parallelism. It works with very low energy consumption. It can store huge amounts of memory in a very small volume. It has made rapid advances in a very brief time frame. Systems are now being proposed that do not use vast amounts of DNA, or require phenomenal error rates, or use expensive reagents. Finally, there are many potential architectures available, combining different biochemical technologies and computational models.

Reasons for pessimism still remain, however. The DNA reactions are slow, taking as much as an hour each. No practical application that fits this technology has been identified. Achieving the desired error rates in the chemical reactions is a major technical challenge. Finally, there is very strong competition from conventional electronic computers, in which phenomenal intellectual and financial investments have been made for more than 40 years.

4 Technologies

4.1 Competing Technologies

To be of practical significance, biomolecular computing systems will need to outperform other kinds of computing systems.

Conventional Electronic Computers Tilak Agerwala, IBM, gave a presentation describing the current state-of-the art in high-performance electronic parallel computing systems, as well as projections for where such systems will be in the next 15 years, based in large part on the 1995 Petaflops Workshop.

Although various proposals for innovative architectures exist, the mainstream view of the computer of the future is that it will utilize standard technology component microprocessors and operating systems with custom interconnection networks and judiciously chosen high-performance system services. Processor speeds currently are in the $^1/_4$ to $^1/_2$ gigaflop[1] range and are likely to increase by a factor of 20 in the next few years. Switch technology will improve at roughly the same rate. By 1998, multi-terafiop machines will be available at a cost of about $100 million. By the year 2015, petaflops systems with 5-10 thousand processors, each with up to 200 gigabytes of memory and 20 terabytes of on-line storage will be available. Of course, special-purpose dedicated computing systems with even better performance characteristics will be possible by then.

Richard Watson, Lawrence Livermore National Laboratory, discussed future trends and needs in high-performance mass storage systems. This arena has traditionally experienced a slower improvement rate in device performance compared to semiconductor processors and memories, and this trend is likely to continue in the future. Thus the performance bottleneck in future systems is likely to be input/output performance. Nevertheless, by the year 2000 systems storing as much as 1 petabyte of data transferred at up to 100 megabytes/sec will be available.

[1] Giga means 10^9, tera means 10^{12}, peta means 10^{15}. Flops are floating-point operations per second.

Other Technologies James Ellenbogen, Mitre Corporation, presented an overview of other technological approaches for building computing devices. Among the most revolutionary of these are molecular-scale nanomechanical computing systems and quantum computers based on interferences among coherent quantum waves. The best-developed technologies, however, are those connected with developing nanometer-scale electronic devices. Such devices include quantum dot cells, quantum-effect solid-state devices and molecular electronic circuit arrays. Quantum dots, for example, are small molecular "boxes' that can hold electrons. These dots can be used to make two-state cells that can be combined into chains, which can convey information without current flow. The cells can also be combined to form logic gates. Since such devices are electronic, they will be easier to integrate with conventional microelectronic devices and hence will be readily adopted and will receive large financial investments. Such technologies will pose stiff competition for biomolecular computing systems.

4.2 Biomolecular Technologies

Research in various biotechnologies may provide useful components in building molecular computers. Rob Lipshutz, Affymetrix, described work being done to construct ordered oligonucleotide arrays using photolithography to deposit DNA on glass. With their technique they can create, on a single glass chip, a grid of cells (called features) in each of which are many copies of a particular DNA strand. They currently are working on feature sizes as small as 10 μm, allowing more than a million features to be stored on a single chip. Thus, for example, they can create an array containing every possible DNA 10-mer on one glass chip. The features can be detected using fluorizine and a scanning confocal microscope. In essence, they have an addressable high-density storage array for DNA. Similar technology using inkjets instead of photolithography is being studied at the University of Washington.

Lloyd Smith, University of Wisconsin, discussed how solid-support chemistry can be used for DNA computing. The advantages of using solids compared to solutions are easy handling, reduced interference between oligonucleotides and easier purification. The disadvantages are limited surface area, decreased hybridization kinetics and surface-chemistry effects. They have designed a system that supports four computational primitives: mark a particular DNA strand (by hybridizing); unmark a strand (by dehybridizing); destroy a marked strand (using exonuclease); and append a sequence to the end of another sequence (using polymerase extension and ligation).

Erik Winfree and Nickolas Chelyapov, Laboratory for Molecular Science at USC, described their work in designing and implementing the above-mentioned stickers model of DNA computing. In order to increase the binding accuracy of the sticker strands they have used peptide nucleic acid (PNA) strands for the stickers. The major bottleneck in their system currently is achieving highly accurate separations.

4.3 Computational Issues

Richard Karp, University of Washington, presented a summary of the computational power of DNA computing. The basic approach of molecular computing to solving NP-hard problems has been to construct a DNA molecule for each potential solution and then use

molecular operations to eliminate invalid solutions.

There are five basic operations used in DNA computing. The *extract* operation separates a tube T into two tubes: one with all molecules containing a particular substring; and another with the remaining molecules. The merge operation simply mixes two tubes. The *detect* operation simply checks if there are any strands in a tube. The *copy* operation amplifies all the strands in a tube. Finally, the *append* operation attaches a given string to the end of every molecule in a tube.

A length-n bit string can be encoded effectively as a DNA molecule. First assign two sequences: one to represent a 0-bit, the other to represent a 1-bit. Then concatenate these patterns separated by sequences that identify the position of each bit in the string. A tube containing all possible such sequences can be generated using copy, append and merge operations.

Lipton showed that using extract, merge and detect operations, the satisfiability of an *n*-variable Boolean formula of size *s* could be accomplished in $O(s)$ operations using tubes with $O(2^n)$ molecules. Adding the append operation allows testing the satisfiability of a size-*s* *n*-variable Boolean circuit within the same bounds. Boneh, Dunworth & Lipton showed that if the circuit represents a one-to-one function *f*, then in $O(s)$ steps a tube can be created containing all strings of the form $(x, fx(z))$. To compute $f^{-1}(y)$ one need only extract the sequence ending in *y* and then sequence it to determine *x*. This is the basis of breaking DES.

Two harder operations to perform are *intersect* and *complement.* Intersect takes two tubes and creates a new tube containing the strings that were contained in both. Complement takes a tube and creates a tube containing the strings that are not in the first tube. If in addition either of these two operations is allowed, then any problem solvable by a conventional computer using a polynomial amount of storage can be solved in a polynomial number of molecular steps.

The strength of DNA computing is its high parallelism. The weaknesses are that the operations are slow and there is no communication within the tube. It is a hard problem, for example, to determine whether a tube contains two identical strands.

The limiting factor has been the volume of DNA that can be used. It takes 0.5 gram of DNA to make 2^{56} strands each of length 1000. To make 2^{70} strands of the same length takes 8 kilograms of DNA. Bach and Condon have given an approach that reduces the number of strands by building the potential solution strands progressively, eliminating impossible solutions before they are fully constructed. Another approach (not guaranteed to find the optimal solution) is to generate only a sample of potential solution strands. Next identify the best strands, discarding the rest, and mutate some of the best strands. Then repeat.

Another limiting factor in DNA computing has been the problem of errors during the extract operation. Karp, Kenyon & Waarts have shown that to achieve separations with a desired error rate of δ using an extract operation with an intrinsic error rate of e requires and can be done with $\theta\left(\lceil \log_{\varepsilon} \delta \rceil^2\right)$ operations.

5 Conclusion and Recommendations

Near its conclusion, the workshop separated into two groups: one consisting of those most interested in computational issues, the other of those interested in biotechnology issues. In the final plenary session of the workshop, each group presented its observations and recommendations.

5.1 Issues of Interest to Computer Scientists

The computer science group prepared a long list of issues of interest to them, organized into five major areas:

Policy Issues In terms of science policy, the main question is what level of investment should be made in these technologies, and in particular, which specific problems and areas are most worth pursuing now. An important issue will be to determine milestones in order to identify when the project is finished (either successfully or not).

Long-Term Goals The first item on the list of long-term goals toward which biomolecular research should aim is, of course, the goal of solving a real-world problem better, either faster or more economically, than conventional electronic computers. Such a "killer application" does not seem to be achievable in the immediate future, but the preliminary work in breaking DES offers an example of a direction that might prove fruitful. Another goal of the project should be the development of technologies that will have uses in biotechnology other than just molecular computing. Such spinoffs will be a very significant benefit of this research. Other goals include obtaining a better understanding of natural computation, *i.e.*, characterizing more precisely what computations are being performed within living cells, and along with that using cells as components in computing systems.

Application Domains The application areas toward which this technology should be targeted include, of course, virtually all aspects of molecular biology. Specific applications where effective molecular computing algorithms might be designed include DNA sequencing, molecular synthesis, drug design and related biological search problems. This research also will benefit the foundations of computer science and the development of new models of computation including those that occur naturally in living cells. Direct applications in information storage and retrieval and the development of new biochemical processes are also indicated.

Technical Issues The most pressing technical issues to be pursued include:

- Characterizing existing biochemical operations as computational primitives —what biochemical steps can be performed and how efficient and reliable is each. This should include the creation of reliable simulation models of the chemical characteristics of these processes so that new experiments can be tested without expensive and time-consuming laboratory work.

- Designing new biochemical tools and techniques.

- Developing new models of computation.

- Developing error-correction techniques both for the final products of the reactions as well as in the biochemical processes themselves, just as living cells repair their defective molecules.

- Improving techniques for input/output to the biomolecular computer — better techniques to create/assemble the molecules that represent a problem instance and to detect/decipher the molecules that represent solutions to the problem.

- Developing hybrid computers with both biomolecular and conventional electronic components.

- Bionanofabrication — using nanotechnology to create and manipulate biocomputing molecules and processes.

- Designing asynchronous chemical computers where the operations/reactions occur continually as they pass through different stages of the system, much like a chemical refinery.

Techniques More specific steps and techniques that would aid the development of molecular computing include:

- Creating simple biomolecular computing components such as registers, adders, stacks, etc.

- Creating mechanisms for specific molecules to communicate with other specific molecules during reactions.

- Developing "garbage collection" techniques to remove unwanted molecular fragments.

- Using a molecular basis for the computation other than DNA, *e.g.*, oligosaccharides, artificial molecules, host-guest chemistry, marking/methylation.

- Developing local-search algorithms that are particularly suited to molecular computation.

- Designing useful subroutines with efficient biomolecular implementations that could be incorporated into larger algorithms.

- Designing analogue DNA computers.

5.2 Issues of Interest to Molecular Biologists

The biotechnology group identified the following areas of interest for further investigation:

- Enzymology. There is a need to characterize better the properties of enzymes that manipulate DNA, particularly their error rates. Better techniques for amplification also need to be developed. This would be useful both for molecular computing and biological applications.

- Surface chemistry. Further efforts are needed to study the use of solid support techniques to interface with DNA.

- Separation. Significant improvements are needed in the accuracy of separating specific molecules from others in a test tube. The challenge here is so great that alternatives, such as the destroy operation, should be explored.

- Errors. Understanding and overcoming the error rates inherent in chemical reactions.

- Self-assembly. Designing reaction systems that would run without external intervention. The computational behavior of such systems, viewed as cellular automata, needs to be better understood.

- Compartmentalization of DNA computing systems. Living cells have specialized components for specific biomolecular tasks. It would be useful to design similar modularization into molecular computing devices.

- Secondary and tertiary structures. It may be possible to use these as part of the computing process, as opposed to current approaches that seek to avoid their effects.

- Biological applications of biomolecular computing. Rather than concentrating on building general-purpose molecular computing, the best payoff may come from designing special-purpose molecular computing processes for specific biological problems, *e.g., in vi*tro selection.

5.3 Conclusions

The consensus of the workshop was that biomolecular computing has great promise in its ability to process enormous quantities of information efficiently. In the very short time since Adleman's original paper, vast strides have been made in better understanding the capabilities of biomolecular computing systems and in overcoming technical difficulties. The notion of biomolecular computing has expanded from just a narrow view of DNA computing to encompass the entire concept of biological information processing. Although no general-purpose DNA computer that will be competitive with conventional electronic systems is likely in the near term, the wide variety of potential architectures for biomolecular systems make further research desirable. Moreover, work in this area will likely have additional impact on other biomedical applications.

GENERAL NSF INFORMATION

The Foundation provides awards for research and education in the sciences and engineering. The awardee is wholly responsible for the conduct of such research and preparation of the results for publication. The Foundation, therefore, does not assume responsibility for the research findings or their interpretation.

The Foundation welcomes proposals from all qualified scientists and engineers and strongly encourages women, minorities, and persons with disabilities to compete fully in any of the research and education related programs described here. In accordance with federal statutes, regulations, and NSF policies, no person on grounds of race, color, age, sex, national origin, or disability shall be excluded from participation in, be denied the benefits of, or be subject to discrimination under any program or activity receiving financial assistance from the National Science Foundation.

Facilitation Awards for Scientists and Engineers with Disabilities (FASED) provide funding for special assistance or equipment to enable persons with disabilities (investigators and other staff, including student research assistants) to work on NSF projects. See the program announcement or contact the program coordinator at (703) 306-1636.

Privacy Act. The information requested on proposal forms is solicited under the authority of the National Science Foundation Act of 1950, as amended. It will be used in connection with the selection of qualified proposals and may be disclosed to qualified reviewers and staff assistants as part of the review process; to applicant institutions/grantees; to provide or obtain data regarding the application review process, award decisions, or the administration of awards; to government contractors, experts, volunteers, and researchers as necessary to complete assigned work; and to other government agencies in order to coordinate programs. See Systems of Records, NSF 50, Principal Investigators/Proposal File and Associated Records, and NSF-51, 60 Federal Register 4449 (January 23, 1995), Reviewer/Proposal File and Associated Records, 59 Federal Register 8031 (February 17, 1994).

Public Burden. Submission of the information is voluntary. Failure to provide full and complete information, however, may reduce the possibility of your receiving an award.

The public reporting burden for this collection of information is estimated to average 120 hours per response, including the time for reviewing instructions. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Gail A. McHenry, Reports Clearance Officer, Information Dissemination Branch, National Science Foundation, 4201 Wilson Boulevard, Suite 245, Arlington, VA 22230.

The National Science Foundation has TDD (Telephonic Device for the Deaf) capability, which enables individuals with hearing impairment to communicate with the Foundation about NSF programs, employment, or general information. To access NSF TDD, dial (703) 306-0090; for FIRS, 1-800-877-8339.

NSF 97-168
(***Electronic Dissemination Only***)